

UNCLASSIFIED

Defense Technical Information Center Compilation Part Notice

ADP012330

TITLE: Coalition Agents Experiment: Multi-Agent Co-operation in an International Coalition Setting

DISTRIBUTION: Approved for public release, distribution unlimited
Availability: Hard copy only.

This paper is part of the following report:

TITLE: KSCO 2002: Second International Conference on Knowledge Systems for Coalition Operations

To order the complete compilation report, use: ADA402533

The component part is provided here to allow users access to individually authored sections of proceedings, annals, symposia, etc. However, the component should be considered within the context of the overall compilation report and not as a stand-alone technical report.

The following component part numbers comprise the compilation report:

ADP012330 thru ADP012354

UNCLASSIFIED

Coalition Agents Experiment: Multi-Agent Co-operation in an International Coalition Setting

David N. Allsopp¹, Patrick Beautelement¹, Jeffrey M. Bradshaw², Edmund H. Durfee³, Michael Kirton¹, Craig A. Knoblock⁴, Niranjan Suri², Austin Tate⁵, Craig W. Thompson⁶

1. QinetiQ Ltd,
Malvern Technology Centre,
St Andrews Road, Malvern,
Worcestershire, WR14 3PS, UK
{d.allsopp, m.kirton}@signal.qinetiq.com, pbeautelement@qinetiq.com

2. Institute for Human & Machine Cognition
University of West Florida
40 South Alcaniz Street
Pensacola, FL 32501, USA
{jbradshaw, nsuri}@ai.uwf.edu

3. EECS Department
University of Michigan
Ann Arbor, MI 48109, USA
durfee@umich.edu

4. Information Sciences Institute
University of Southern California
4676 Admiralty Way
Marina del Rey, CA 90292, USA
knoblock@isi.edu

5. Artificial Intelligence Applications Institute
Centre for Intelligent Systems and their Applications
Division of Informatics, The University of Edinburgh
80 South Bridge, Edinburgh EH1 1HN, UK
a.tate@ed.ac.uk

6. Object Services and Consulting, Inc. (OBJIS)
2725 Deep Valley Trail, Plano, TX 75023, USA
thompson@objis.com

Abstract. Military Coalitions are examples of large-scale multi-faceted virtual organizations, which sometimes need to be rapidly created and flexibly changed as circumstances alter. The Coalition Agents eXperiment (CoAX) aims to show that multi-agent systems are an effective way of dealing with the complexity of real-world problems, such as agile and robust Coalition operations and enabling interoperability between heterogeneous components including legacy and actual military systems. CoAX is an international collaboration carried out under the auspices of DARPA's Control of Agent-Based Systems (CoABS) program. Building on the CoABS Grid framework, the CoAX agent infrastructure groups agents into domains that reflect real-world organizational, functional and national boundaries, such that security and access to agents and information can be governed by policies at multiple levels. A series of staged demonstrations of increased complexity are being conducted in a realistic peace-enforcement scenario situated in 2012 in the fictitious African state of "Binni". These demonstrations show how agent technologies support the rapid, co-ordinated construction of a Coalition command system for intelligence gathering, for visualization, and for campaign, battle and mission planning and execution.

1 Introduction and Background

1.1 Military Context

Success in military operations involves carrying out high-tempo, coherent, decisive actions faster than an opponent can react, resulting in decision dominance through the use of command agility. Command agility is about being flexible and adaptable so that fleeting opportunities can be grasped; the Commander issues clear intent and then delegates control to subordinates, allowing them the scope to exercise initiative. It also means being innovative, creative and unpredictable in a manner that (even if low-tempo) increases confusion in the mind of an opponent. This process is command led; human decision-making is primary and the role of technology is secondary. Shared understanding and Information Superiority are key enablers in this process and are fundamental to initiatives such as the UK's Command and Battlespace Management program, the US Joint BattleSpace Infosphere program and, more generally, Network-Centric Warfare (<http://www.dodccrp.org/>).

In addition to the problems of integrating single-service and Joint capabilities into a coherent force, the nature of Coalition (multi-national) operations implies some need to rapidly configure foreign or ‘come-as-you-are’ systems into a cohesive whole. Many problems in this environment can only be solved by organizational changes and by ‘aligning’ doctrine, concepts of operations and procedures. Due to the inevitable absence of pre-existing co-ordinated systems, Coalition scenarios require a rapid, flexible, on-the-fly approach that allows capabilities to be assembled at run-time. However, in addressing this requirement for interoperability, it is also crucial to address issues of security of data, control over semi-trusted software from other Coalition partners, and robustness of the resulting system (e.g. the ability to withstand denial-of-service attacks).

Currently, Coalition operations are often characterized by data overload, information starvation, labor intensive collection and co-ordination of information, and standalone stove-pipe command systems that use incompatible data formats. This leads to a horrendous technical integration task and gives commanders only scattered snapshots of the battlespace. This paper aims to show that the agent-based computing paradigm offers a promising new approach to dealing with such issues by embracing the open, heterogeneous, diverse and dispersed nature of the Coalition environment. In this paper, we show that software agents that act on behalf of human users enable military commanders to act decisively in cyberspace and thus contribute towards the achievement of ‘Cyberspace Superiority’, a critical component of warfare in the information age (Alberts et al, 2001).

1.2 Software Agent Technology

Software agents are currently receiving much attention in the research community. This interest is being driven by the phenomenal growth of the Internet and the World-Wide-Web. Agents can be viewed as semi-autonomous software designed to help people cope with the complexities of working collaboratively in a distributed information environment. This involves the agents communicating between the users and between themselves. The agents are used to find, format, filter and share information, and work with users to make the information available wherever and whenever they need it. The agents are also able to proactively suggest courses of action, monitor mission progress, and recommend plan adjustments as circumstances unfold.

A community of agents can be seen as a set of distributed, asynchronous processes communicating and sharing information by message passing in some infrastructure. In this regard, an important output from DARPA's CoABS program is the CoABS Grid — a middleware layer based on Java / Jini technology that provides the computing infrastructure to integrate heterogeneous agent communities and systems rapidly (<http://coabs.globalinfotek.com/>).

A recent article (Jennings, 2001) argues that the agent paradigm is a good way of building complex software systems in general, and hence offers potential benefits in the Coalition setting. For example, legacy command systems could be provided with software agent wrappers that allow them to inter-operate and share information with other systems and agent applications in a loosely connected, heterogeneous architecture, underpinned by the CoABS Grid. The scenario used as the basis of the experiments to test this hypothesis is described in section 2.

1.3 Aims of the CoAX Project

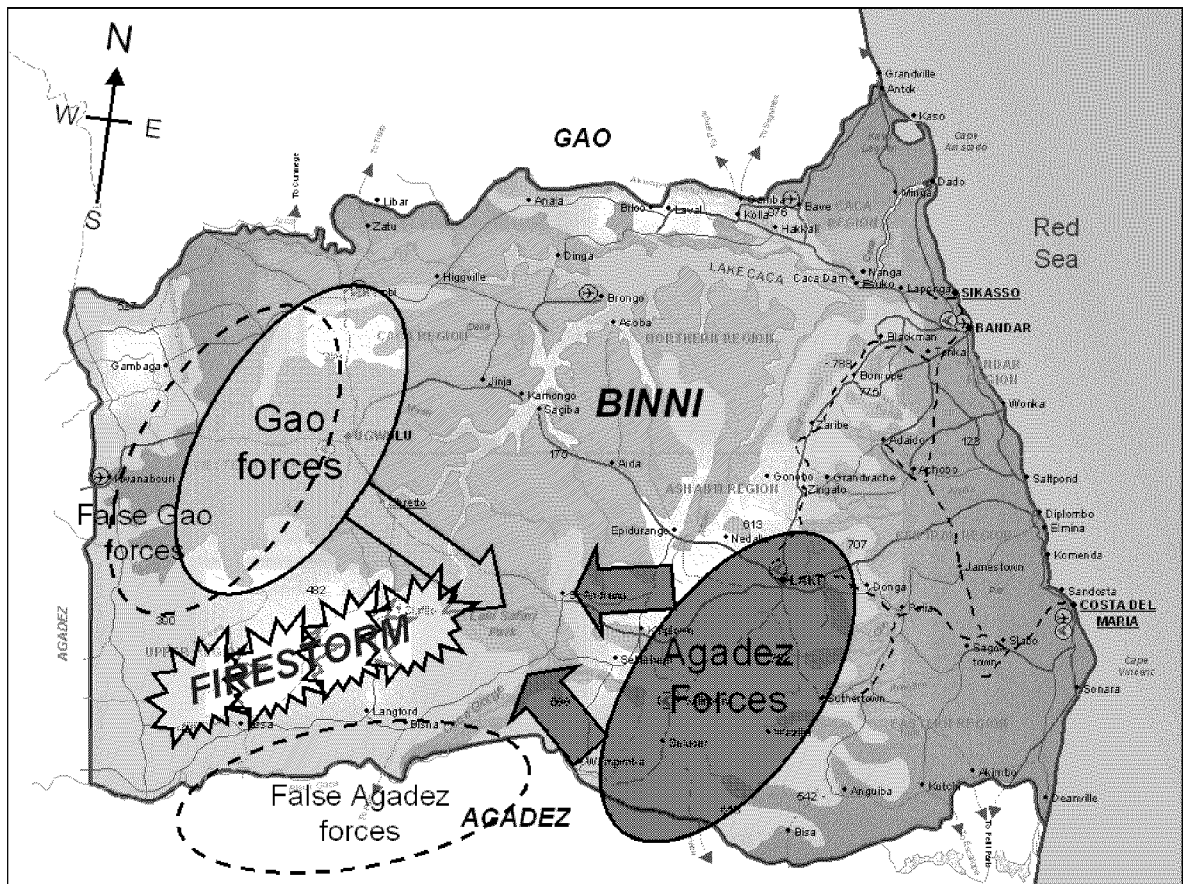
This paper describes the progress of an international collaborative effort whose overall goals are to demonstrate that the agent-based computing paradigm offers a promising new approach to dealing with the technical issues of establishing coherent command and control (C2) in a Coalition organization. This collaborative effort, entitled CoAX (Coalition Agents eXperiment), is a Technology Integration Experiment under the auspices of DARPA's Control of Agent Based Systems (CoABS) program (<http://www.aiai.ed.ac.uk/project/coax/>). Specific hypotheses of the research program are that:

- agents are a useful metaphor for dealing with the complexity of real-world systems such as military operations;
- an agent-based C2 framework can support agile and robust Coalition operations;
- software agents can be used to enable interoperability between legacy or previously incompatible systems;
- the CoABS Grid can be used to rapidly integrate a wide variety of agents and systems — i.e., rapid creation of virtual organizations;
- domain policies can structure agent relationships and enforce Coalition policies;
- intelligent task and process management can improve agent collaboration;
- semantic web technology can improve agent interoperability between disparate Coalition command systems.

The CoAX team has built a software agent test-bed based on the CoABS Grid (<http://coabs.globalinfotek.com/>). This paper describes the work done, the demonstrations carried out so far, the scenario and storyboard used and some of the insights gained.

The paper begins by describing the Coalition scenario and military command structure used in our demonstration experiments. Section 3 describes the corresponding agent architecture that was developed to reflect the military organizational structure. The events occurring in the storyboard used for the various demonstrations so far are described in Section 4. A preliminary assessment of software agent capabilities and a discussion of future research are provided in Section 5. Concluding remarks are given in Section 6.

The CoAX work needed a suitably realistic scenario for its experiments and so we expanded the fictional "Binni" scenario (Rathmell, 1999) developed for The Technology Co-operation Programme. In this scenario the year is 2012 and global warming has altered the political balance of the world. The action is set in an area that is currently the Sudanese Plain (Figure 1). Previously uninhabited land in the Plain is now arable and the area has received large amounts of foreign investment. It is now called "The Golden Bowl of Africa".



A conflict has developed between two countries in the area. To the north is Gao, which has expansionist aspirations but which is only moderately developed, with old equipment and with a mostly agrarian society. To the south is Agadez, a relatively well developed and fundamentalist country. Gao has managed to annex an area of land, called it Binni and has put in its own puppet government. This action has come under fierce attack from Agadez. Gao has played the ‘threat of weapons of mass destruction from Agadez’ card and has enlisted support from the UN who have deployed a force, the UN War Avoidance Force for Binni (UNWAFB), to stabilize the region. This basic scenario was adapted for a number of CoAX demonstrations (see Section 4), beginning with the initial planning phase, then moving onto shorter timescales and more dynamic, uncertain events for the execution phase.

This Binini Coalition operation needs to rapidly configure various incompatible, ‘come-as-you-are’ or foreign systems into a cohesive whole within an open, heterogeneous and dispersed environment. This scenario provides a

suitable test for the software agent experiments, where run-time composability is a very close metaphor for the dynamic uncertainty of Coalition operations. The complexity of the situation must not be underestimated and is best illustrated by looking at the Binni Coalition Command Structure shown in Figure 2 below.

This is a representative and realistic Coalition command structure involving the UN, Governments, Other Government Departments (OGDs, such as the Foreign Office), Non-Government Organizations (NGOs, such as Oxfam), representatives of all the Coalition countries (with their own ‘ghosted’ Command Structures) and the Coalition HQs and subordinate fighting forces. The solid black lines on the diagram show the legal lines of authority (the command chain) and accountability. This is the kind of Coalition structure that would be agreed by the participants; no part of the formal command chain is owned by any specific country. Note that the ‘levels of command’ overlap and their boundaries are not rigidly defined. Dashed lines show an advisory / negotiating role.

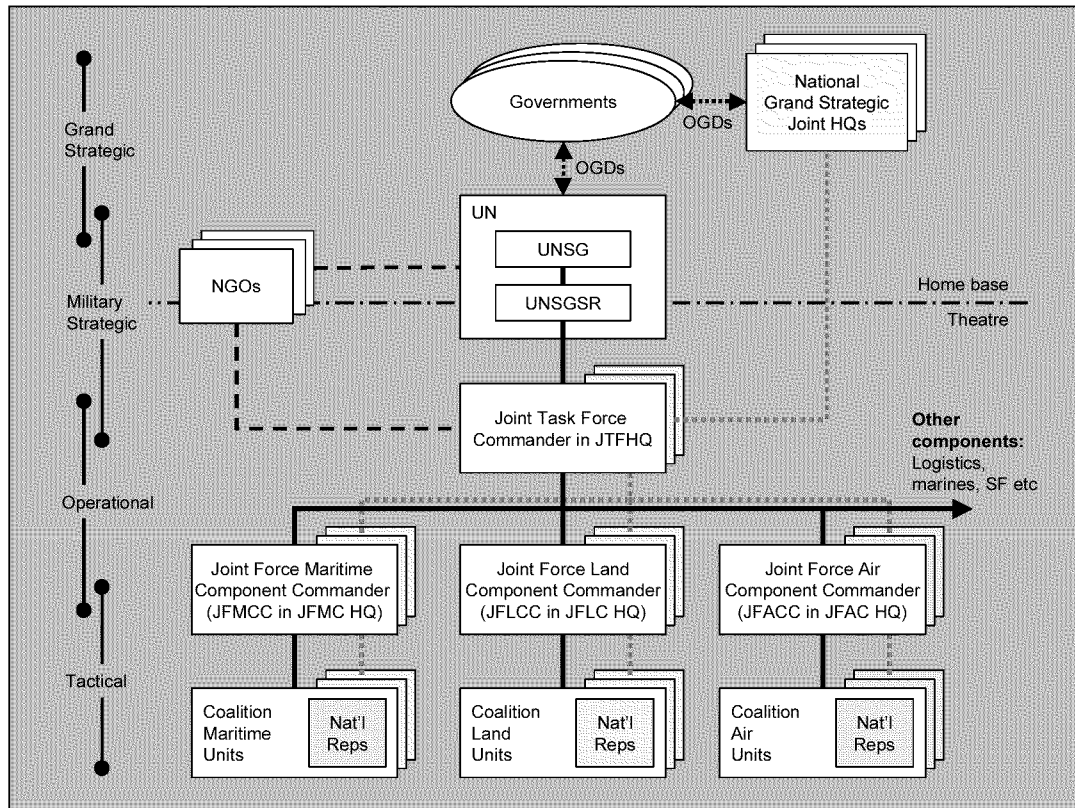


Figure 2: A representative Coalition structure, showing the chain of command down from the United Nations, including the ‘ghosted’ command structures of the participant nations, and Non-Government Organizations (NGOs). The approximate command level at which the various entities operate is indicated on the left.

3 Software Agent Architecture

3.1 Human Domains

Integrating information across a Coalition is not just a matter of employing technology — it involves the creation of a coherent ‘interoperability of the mind’ at the human level as well, where many social and cultural factors come into play. The mapping between the human and technical worlds is thus not straightforward. From the human perspective, we identified four kinds of ‘domains’:

- **Organizational Domains:** for example the Joint Task Force HQ (JTF HQ) ;
- **Country Domains:** each of the National command chains would be a separate, self-contained domain;
- **Functional Domains:** sets of entities collaborating on common tasks, for example Meteorology or Intelligence ;
- **Individual Human Domains of Responsibility:** Commanders have responsibility for their own HQ and all subordinate ones (in practice they delegate). Hence the individual human domains of influence may overlap.

These types of domains are not entirely exclusive and there are many different levels of overlap and interaction depending on the viewpoint taken. It is this complexity at the human level that creates difficulties for technical systems.

3.2 Software Agent Domains

3.2.1 CoABS Grid Infrastructure

At the most basic level, the agents and systems to be integrated require infrastructure for discovery of other agents, and messaging between agents. The CoABS Grid provides this. Based on Sun's "Jini" services which are themselves based upon Java's Remote Method Invocation, the Grid allows registration and advertisement of agent capabilities, and communication by message-passing. Agents on the Grid can be added or removed, or their advertisements updated, without reconfiguration of the network. Agents are automatically purged from the registry after a short time if they fail. Multiple lookup services may be used, located by multicast or unicast protocols. In addition, the Grid provides functionality such as logging, visualization, and more recently encryption of messages and agent authentication.

3.2.2 KAoS Domain Management

The increased intelligence afforded by software agents is both a boon and a danger. By their ability to operate independently without constant human supervision, agents can perform tasks that would be impractical or impossible using traditional software applications. On the other hand, this additional autonomy, if unchecked, also has the potential of effecting severe damage to military operations in the case of buggy or malicious agents. The Knowledgeable Agent-oriented System (KAoS) provides services that help assure that agents from different developers and running on diverse platforms will always operate within the bounds of established policies and will be continually responsive to human control so that they can be safely deployed in operational settings (Bradshaw et al., 1997, 2001). KAoS services and tools are intended to allow for the specification, management, conflict resolution, and enforcement of policies within the specific contexts established by complex military organizational structures.

KAoS domain management services can be used to group agents into logical domains corresponding to organizational structures, administrative groups, and task-oriented teams. Within CoAX, these domains mirror the human domains described above, allowing for complex hierarchical, heterarchical, and overlapping structures. An agent domain consists of a unique instance of a domain manager (DM) along with any agents that are registered to it. Alternatively, an intensionally-defined domain consists of a set of agents sharing one or more common properties (e.g., the domain of all agents physically residing on some host). The function of a domain manager is to manage agent registration, and serve as a single point of administration and enforcement for domain-wide, host-wide, VM-wide, VM-container-wide, or agent-specific policies.

3.2.3 Domain policies

A policy is a declarative constraint governing the behavior of one or more agents, even when those agents may not be domain-aware or where they may be buggy or malicious. For example, a policy may be declared that all messages exchanged among agents in the JFAC HQ domain must be encrypted, or that an agent cannot simultaneously belong to the US and the UK domain. A policy does not tell the agent how to perform its task; it rather specifies the conditions under which certain actions can be performed. By way of an analogy to traffic management, it is more like a set of individually-customizable stop signs and highway patrol officers that define and enforce the rules of the road than it is like a route planner that helps agents find their way to their destinations.

Policies governing authorization, encryption, access control, and resource control are part of KAoS domain management. However, due to our focus on agent systems our scope goes beyond these typical security concerns in significant ways. For example, KAoS pioneered the concept of agent conversation policies (Bradshaw et al., 1997). Teams of agents can be formed, maintained, and disbanded through the process of agent-to-agent communication using an appropriate semantics. In addition to conversation policies, we are developing representations and enforcement mechanisms for mobility policies, domain registration policies, and various forms of obligation policies. These policies are represented in ontologies using the DARPA Agent Markup Language (DAML), and an efficient description logic-based approach is used as the basis for much of the domain manager's reasoning to discover and resolve policy conflicts and to perform other kinds of policy analysis.

The separation of policy specification from policy-enforcement mechanisms allows policies to be dynamically re-configurable, and relatively more flexible, fine-grained, and extensible. Agent developers can build applications whose policies can change without necessarily requiring changes in source code. The rationale for using declarative policies to describe and govern behavior in agent systems includes the following claims: easier recognition of non-normative behavior, policy reuse, operational efficiency, ability to respond to changing conditions, and the possibility of off-line verification.

3.3 Software Agent Domains in CoAX

The CoAX demonstrations contain software agents grouped into agent domains using the CoABS Grid, with the policies enforced by KAOs domain management services. A typical domain configuration is shown in Figure 3.

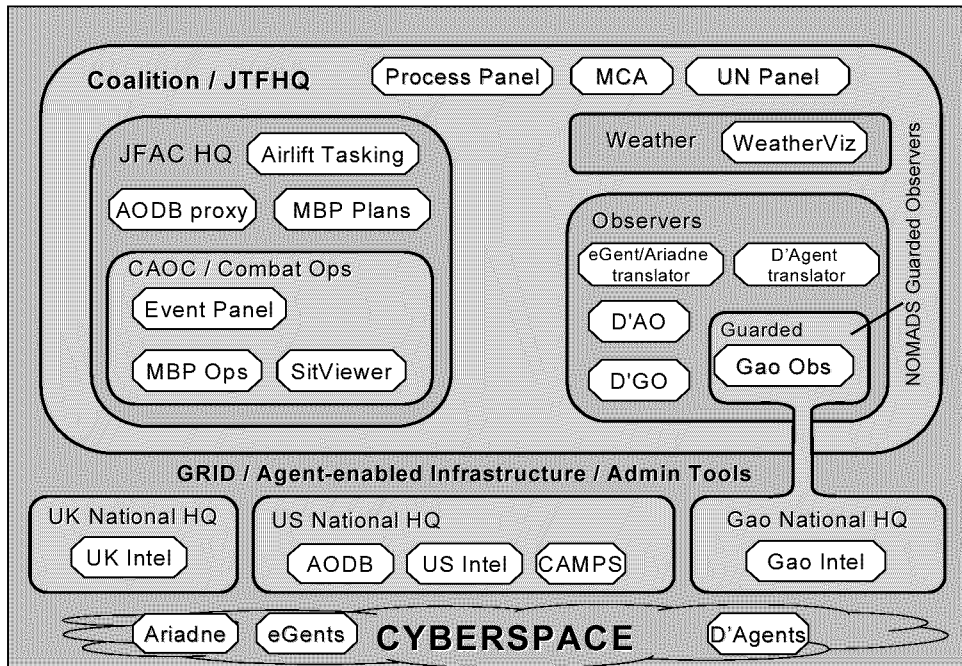


Figure 3. Typical CoAX domain structure; domains are indicated by rounded rectangles; agents by angled rectangles. Some agents are proxies for agents or legacy systems that are not themselves domain aware. Each domain would also contain a Domain Manager agent and a Matchmaker agent (omitted for clarity). Nesting of domains indicates a hierarchy of responsibility and policy control. The agent acronyms are expanded in the body text.

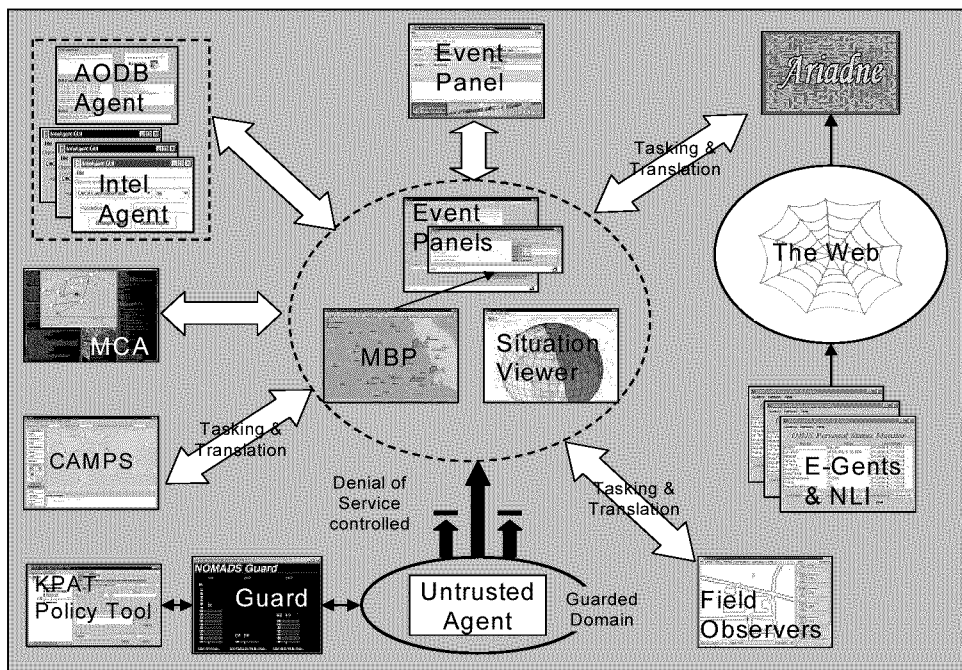


Figure 4: Overview of technologies and agents. The central visualization and planning tools find and acquire data (e.g. disposition of ground forces) and services (e.g. airlift scheduling and plan deconfliction) from the other agents and systems, in some cases via intermediate tasking and translation agents. MBP = Master Battle Planner, MCA = Multi-level Coordination Agent, KPAT = KAOs Policy Admin Tool, AODB = Air Operations Data Base, NLI = Natural Language Interface, CAMPS = Consolidated Air Mobility Planning System.

4 Demonstration Storyboard and Technologies

In this section we progress through the storyboard created for the Binni Scenario, and describe each of the agent systems and technologies brought into play for each part of the story. An overview of the interactions from the agent/system point of view is shown in Figure 4.

4.1 Population of Domains

Following the outbreak of hostilities, the UN has deployed their UN War Avoidance Force for Binni (UNWAFB), to stabilize the region. The active Coalition participants at this time are the UK, US and Gao.

In agent terms, a variety of agent domains are set up using the CoABS Grid infrastructure and the KAoS domain management services, representing the organizational structures (the JTF HQ and the JFAC HQ), the nations (UK, US, Gao) and various functional domains such as Weather and Observers. These domains are populated with a number of agents, which register with their Domain Manager and optionally advertise their services with their domain Matchmaker.

4.2 Data Gathering and Air Planning

After exploring options to separate the opposing forces and restore the peace in the region, the deployment of a large ground observation and peace enforcement force and other courses of action have been rejected, and a 'Firestorm' mission has been decided upon. This will clear land to enable simpler remote and ground observations with less risk to the Coalition peacekeepers. The Coalition undertakes initial information gathering and planning.

4.2.1 Master Battle Planner (MBP)

Air planning at the JFAC is performed using QinetiQ's MBP, a highly effective visual planning tool for air operations. MBP assists planners by providing them with an intuitive visualization on which they can manipulate the air intelligence information, assets, targets and missions, using a map-based graphical user interface (Figure 5). This enables an operator to build a battle scenario containing targets, offensive and defensive units, airspace measures and other objects using simple dialogs and point-and-click techniques on the map. Objects on the map can then be moved around, and their properties can be changed. Information such as the allegiance and status of units, and the ranges of units may also be displayed.

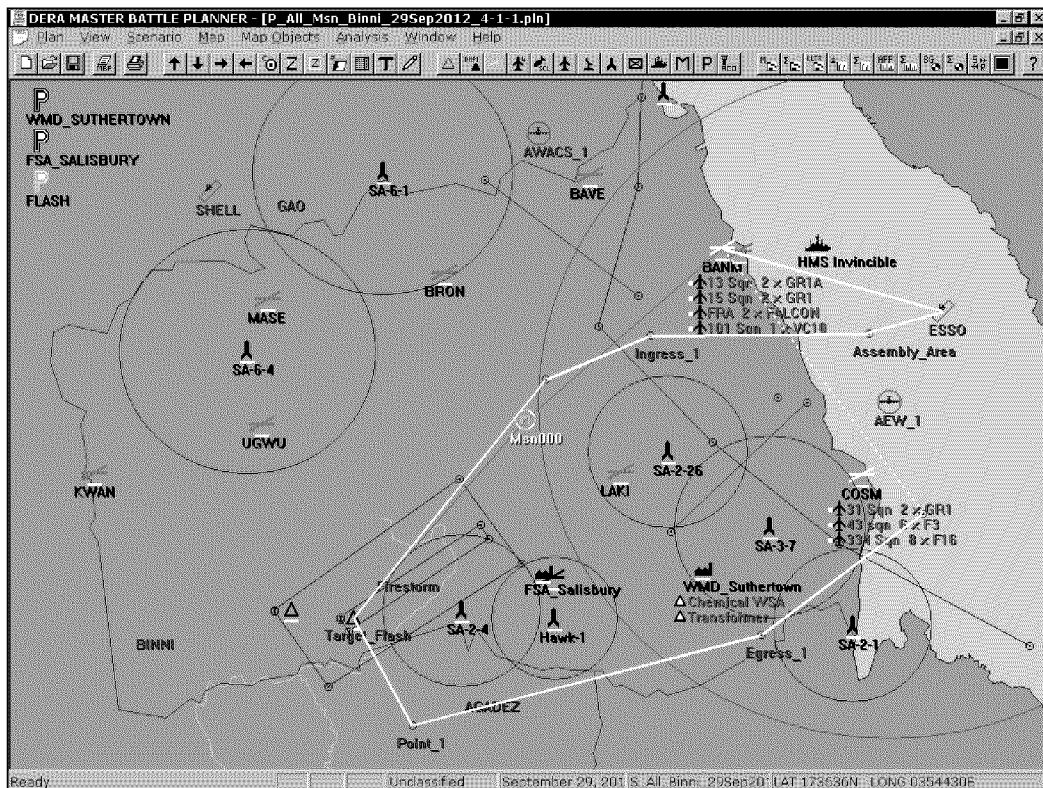


Figure 5: Master Battle Planner map display of the fictional countries of Binni, Gao, and Agadez. A selected mission is highlighted, proceeding from an airbase (BANM), to refueling tanker (ESSO), to the target via waypoints and airspaces, and back to base by a different route.

The operator can interact with these entities and can plan individual air missions (or more complex packages of missions) by dragging and dropping offensive units onto targets on the map. Supporting / defensive elements are added in the same way. The system gives the operator analytical tools to assess the planned air operations for:

- the best utilization of resources; (e.g. by highlighting air units that are over-tasked);
- time-phasing (through charts and animated ‘fly-out’);
- concordance with the military guidance given.

MBP is a monolithic C++ application, which has been agent-enabled by wrapping it in Java code, using the Java Native Interface. The agent-enabling of MBP allows it to receive all the scenario data (targets, assets, airspaces etc.) from multiple information-providing agents (‘Intel Agents’ — see Figure 4) and update this information in near-real time. Importantly, this process is integrated into the normal usage of MBP; when an operator views the status of an object, agents are automatically tasked to update the information. Agents may also ‘push’ changes of status to MBP. Information concerning other air missions can be accepted and merged with missions planned within MBP, as described below. Missions can also be saved and exported, enabling other agents to reason with the data.

4.2.2 Consolidated Air Mobility Planning System (CAMPS)

The second real military system integrated into the demonstration is the Air Force Research Laboratory’s CAMPS Mission Planner (Figure 6). CAMPS develops schedules for aircraft to pick up and deliver cargo within specified time windows. It takes into account constraints on aircraft capabilities, port handling capabilities, crew availability and work schedule rules, etc. Users of the planner develop plans (schedules) for aircraft to carry a particular cargo, specifying the intermediate ports, air refueling tracks and the kinds of crews that will be available. They can also specify a number of constraints on the airports potentially involved in the plans to be developed (Emerson & Burstein, 1999; Burstein et al, 2000).

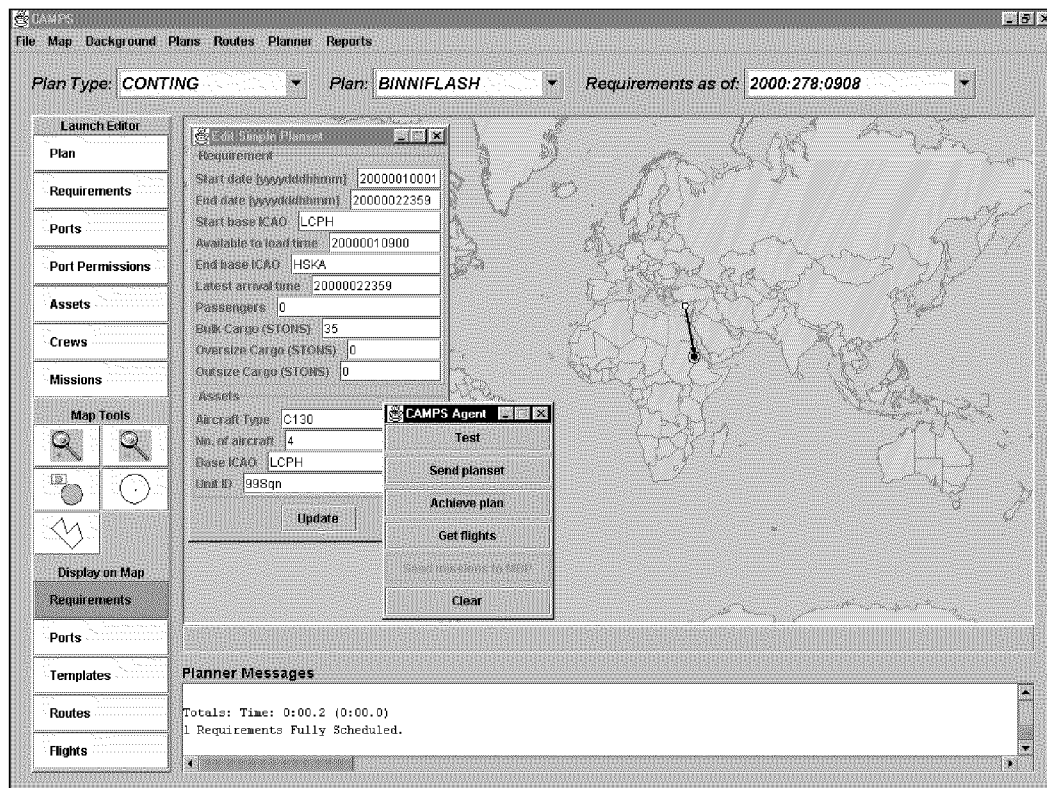


Figure 6: The CAMPS airlift planner, and the demonstration agent used to task the CAMPS agent with a simple requirement: movement of cargo from Cyprus into the fictional country of Binni.

In the demonstration scenario, CAMPS schedules airlifts of cargo into Binni. These airlift flights could conflict with offensive air missions, so the scheduled flights are requested from the CAMPS agent, and sent to MBP, forming part of the normal MBP air visualization. This is achieved by an intermediate agent which tasks CAMPS, and also translates between the KQML messages used by CAMPS and the XML messages used by the MBP agent.

This is an interesting example, as only partial translation is possible; CAMPS and MBP differ fundamentally in their definition of air missions. A CAMPS mission consists of an arbitrary collection of flights, where a flight is a single journey from A to B by a single aircraft. However, an MBP mission consists of a starting point and a route, which must return to the starting point (perhaps by a different path), and may consist of multiple aircraft. CAMPS can therefore produce routes that have no fully valid representation in MBP, although they could be partially represented or indicated graphically.

4.2.3 Ariadne

In a similar manner, weather information extracted from websites by the Ariadne system from the University of Southern California, Information Sciences Institute, is translated and forwarded to MBP, again forming part of the normal picture of the air situation. Ariadne facilitates access to web-based data sources via a wrapper / mediator approach (Knoblock and Minton, 1998). Wrappers that make web sources look like databases can be rapidly constructed; these interpret a request (expressed in SQL or some other structured language) against a web source and return a structured reply. The mediator software answers queries efficiently using these sources as if they formed a single database. Translation of the XML from Ariadne into the XML expected by MBP was handled by custom code, but can now be performed more easily using XSLT (Extensible Stylesheet Language Transformations); this latter technique is used elsewhere in the demonstration (section 4.2.6).

4.2.4 I-X Process Panels (I-P²)

This Coalition planning process is supported using I-X process panels. I-X is a research program with a number of different aspects intended to create a well-founded approach to allow humans and computer systems to cooperate in the creation or modification of some product such as a plan, design or physical entity — i.e. it supports synthesis tasks. I-X may also be used to support more general collaborative activity. The I-X research draws on earlier work on O-Plan (Tate et al, 1998), <I-N-OVA> (Tate, 1996) and the Enterprise Project (Fraser and Tate, 1995) but seeks to make the framework generic and to clarify terminology, simplify the approach taken, and increase re-usability and applicability of the core ideas. Within CoAX, the I-X approach is being used to provide task and process support and event-response capabilities to various Coalition participants (Figure 7).

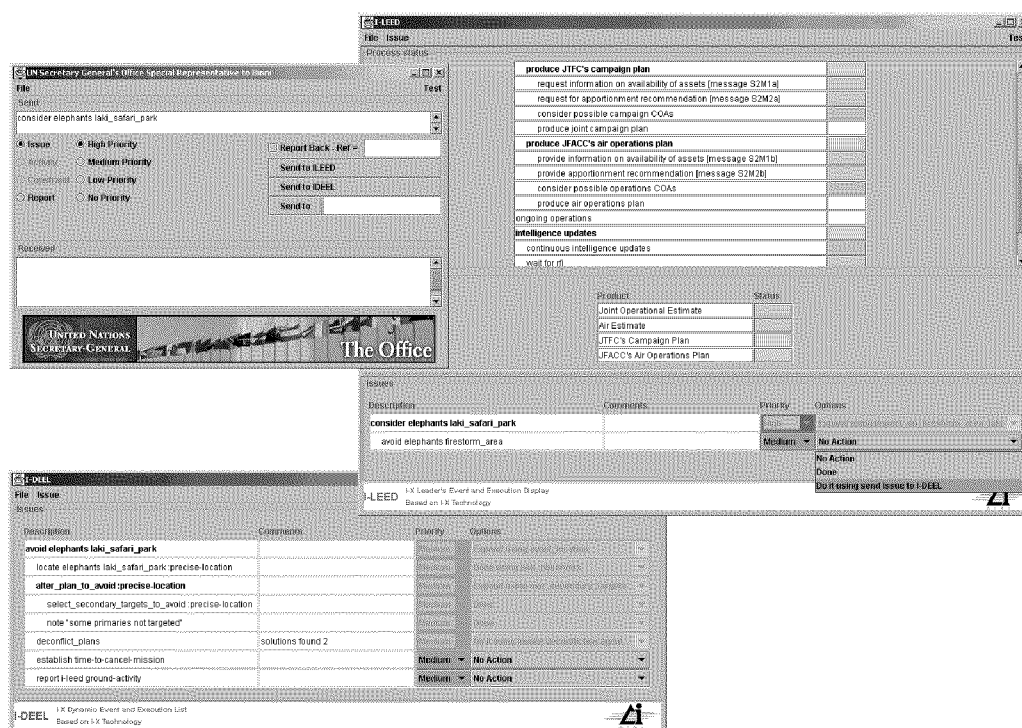


Figure 7: I-X Process and Event Panels

The aim of an I-X Process Panel (I-P²) is to act as a workflow and messaging 'catch all' for its user. It can act in conjunction with other panels for other users if desired. A panel:

- Can take *any* requirement to:
 - Handle an issue;
 - Perform an activity;

- (in future) Add a constraint.
- Deals with these via:
 - Manual (user) activity;
 - Internal capabilities;
 - External capabilities (invoke or query);
 - Reroute or delegate to other panels or agents (pass);
 - Plan and execute a composite of these capabilities (expand).
- Receives reports and interprets them to:
 - Understand current status of issues, activities and constraints;
 - Understand current world state, especially status of process products;
 - Help the user control the situation.
- Copes with partial knowledge.

4.2.5 Resource control via domain policies

Gao has host nation status within the Coalition but its intentions are unclear and it is distrusted. Special steps are taken to monitor the information passed to and from Gao within the Coalition. During the demonstration, misinformation feeds by Gao (intended to displace the firestorm to allow Gao to take an advantage and move forward) are detected and thwarted. Gao becomes belligerent and launches a denial of service attack against the Coalition's C3I infrastructure.

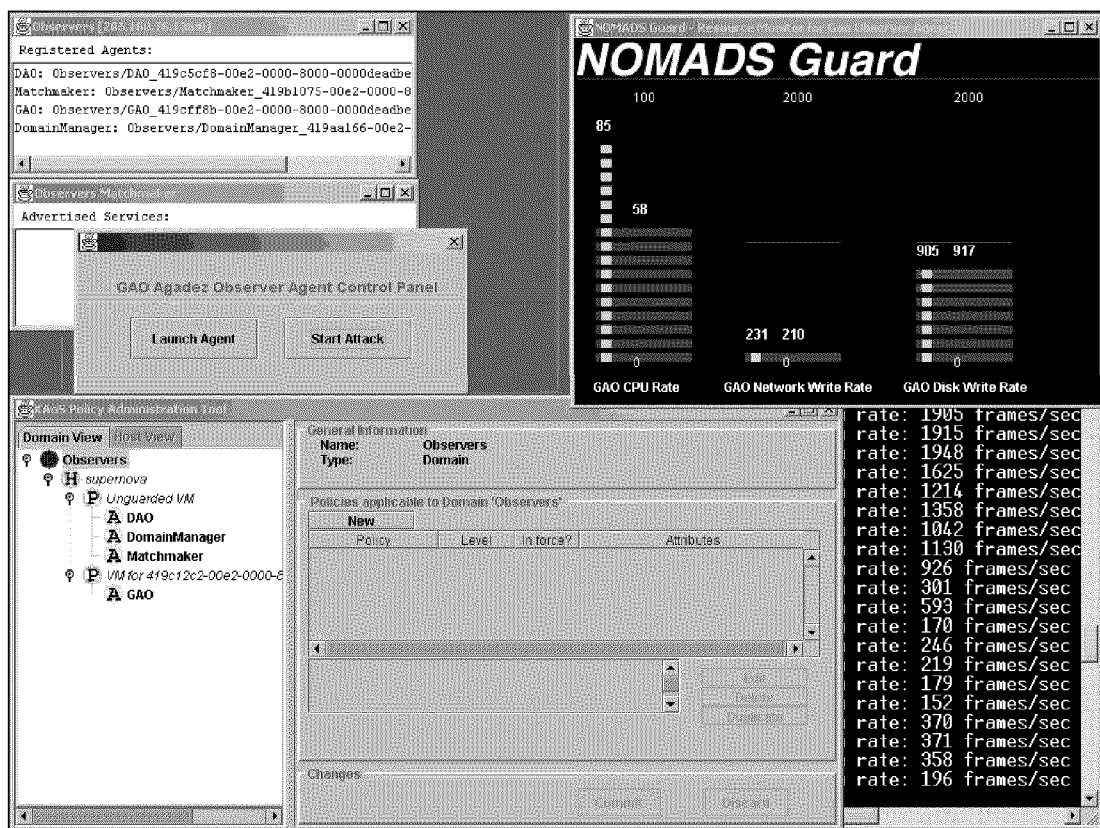


Figure 8: A denial-of-service attack by the Gao agent is starving other agents of resources (note the decreasing rate of processing in the console, bottom right). The Guard (top right) is monitoring the resource usage of the Gao agent. The excessive resource usage triggers a change in domain policy, and the resource limits enforced by the AromaVM are lowered. The policy can also be changed manually using KPAT, the KAoS Policy Administration Tool (bottom left).

The Gao agent in the demonstration is run under NOMADS, a mobile agent system from IHMC. The NOMADS project aims to develop a set of distributed agent-based systems using the Java language and environment. The agent code runs in a new Java Virtual Machine, the AromaVM. The AromaVM provides two key enhancements over standard Java VMs: the ability to capture the execution state of threads and the ability to control resources consumed by threads. By capturing the execution state of threads, the NOMADS agent system provides strong or transparent mobility for agents.

In addition, the resource control mechanisms can be used for controlling and allocating resources used by agents as well as to protect against denial of service attacks by malicious agents. When the Gao agent exceeds certain resource limits, an automatic change in domain policy is triggered by a domain Guard, and the AromaVM is instructed to reduce the resources available to the malicious agent (Figure 8). An operator can manually reduce the limits further, using the KAoS Policy Admin Tool (KPAT).

4.2.6 Data feeds from mobile devices and observers

The firestorm mission has been planned and aircraft have already taken off. However the news media break a story that wildlife in an important safari park in Binni may be in danger as the park overlaps the firestorm area. With only an hour to go, the UN Secretary General's Special Representative to Binni asks the Joint Task Force Commander to consider the wildlife risk aspects of the planned approach. Dynamic information gathering and information feeds using agent technology are employed to create a real time feed of the position of some at-risk large mammals.

This urgent issue is noted and broken down into sub-tasks using the event panels. The progress of aircraft is monitored in near real-time on the Situation Viewer agent from QinetiQ, and the time left before aircraft are committed to their targets is determined from MBP. A search is made for information on the locations of animals in the safari park, and it is discovered that data are available on-line via agents running on monitoring devices attached to large mammals in the park. The agents are eGents (agents that communicate by email) developed by Object Services and Consulting, Inc (OBJS). Historical data from these devices is queried using a Natural Language Interface from OBJS. To aid the planners, a live data-feed is created from the safari park website, using Ariadne to extract data from the pages, and a translator agent using XSLT. The resulting message stream is sent to MBP and to the Situation Viewer agent, allowing the current position and track of the animals to be visualized (Figure 9).

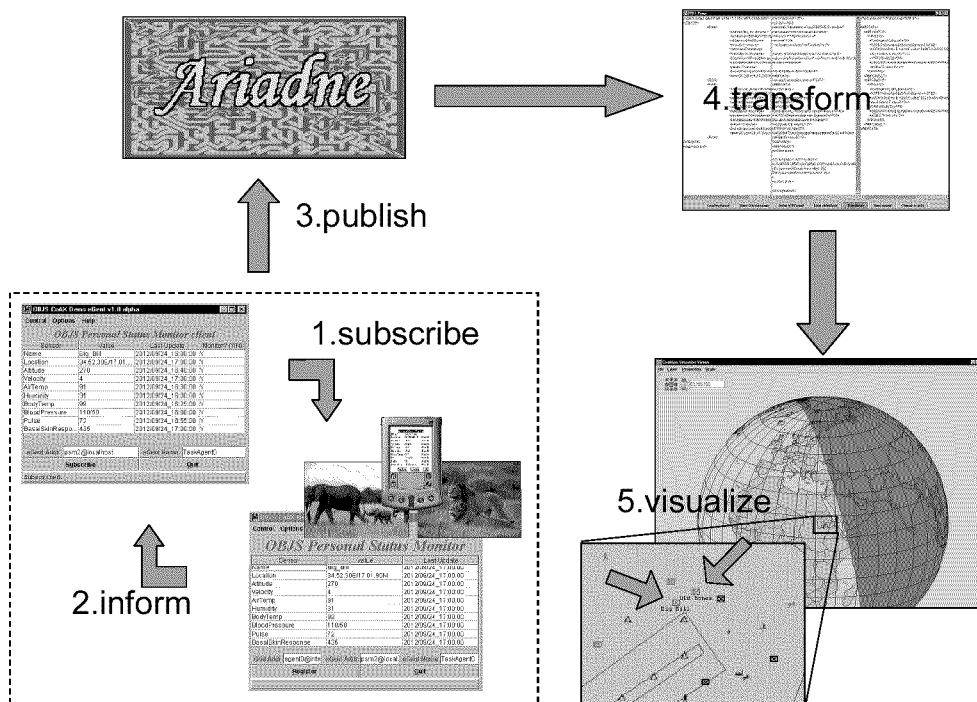


Figure 9: An eGent client subscribes to eGents running on mobile devices (wildlife tags). The data from these devices are published by the client on a web page. Ariadne extracts data from the webpages, and produces XML. The XML is transformed to another format by another agent using XSL Transformations, and finally sent to agents such as MBP and Situation Viewer for visualization.

Data about the movement of ground forces, from the D'Agents field observation system from Dartmouth College, are also transformed using another instance of the translator agent and visualized in the same way, allowing the coalition to identify a convergence of hostile forces on the Laki safari park area.

4.2.7 Plan export and deconfliction

After consideration it is decided to continue with the firestorm mission, but to re-plan as necessary to avoid risk to wildlife. Firestorm targets are adjusted in time or secondary targets selected as necessary for the first wave of firestorm bombing. The impacts of these changes on the Coalition's medical and humanitarian operations are automatically detected, and unintended conflicts between disjoint Coalition operations are avoided.

The air plans are revised using MBP, and then sent to a deconfliction agent to check them against planned activities in other Coalition HQs. The Multi-level Coordination Agent (MCA) from the University of Michigan processes the plans, using multiple levels of abstraction to generate solutions (Clement & Durfee, 1999). The planners are kept informed of progress via their I-X event panels, and can view the results on the MCA display when ready (figure 10). The plans are adjusted iteratively until the conflicts are resolved.

4.2.8 Dynamic Forced Migration (Scram) of Observer Agents

Agadez seeks to use this complication to seize the initiative and launches fighter attacks against a Coalition airborne high value asset (JSTARS) that is monitoring the operation. When this attack is detected, the JSTARS starts to regress, which implies that the observer agents on the JSTARS will not be able to continue providing information to the coalition.

In order to solve this problem, the administrator uses the forced migration (scram) capabilities of the NOMADS mobile agent system to move the observer agents from the JSTARS platform to a secondary ground station platform. The NOMADS system uses the state capture mechanisms in the Aroma VM to capture the full execution state of the agents on the JSTARS. Once captured, the execution state is sent to a new platform where the agents can be restarted without any loss of their ongoing computations (figure 11). This allows the observation agents to continue to operate on the ground station and provide information to the coalition even after the JSTARS regresses.

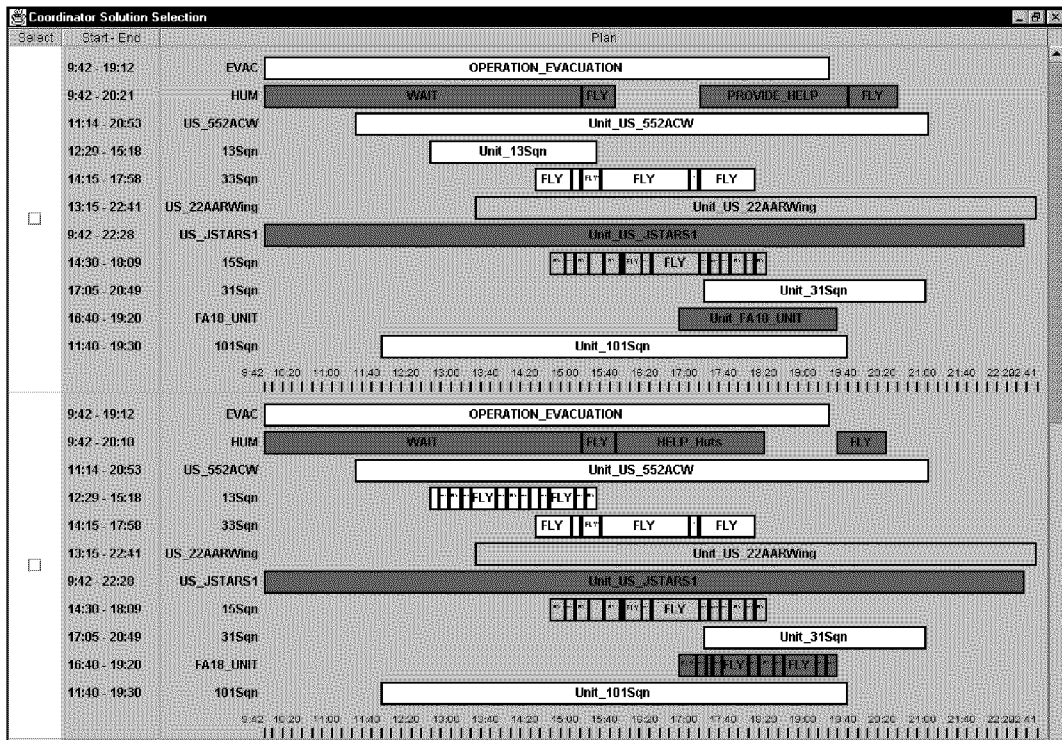


Figure 10: Deconfliction of Coalition plans by the Multi-level Coordination Agent. In the second solution (lower half) two missions (13Sqn and the FA18_UNIT) have been broken down to a lower level of abstraction to seek more optimal coordination

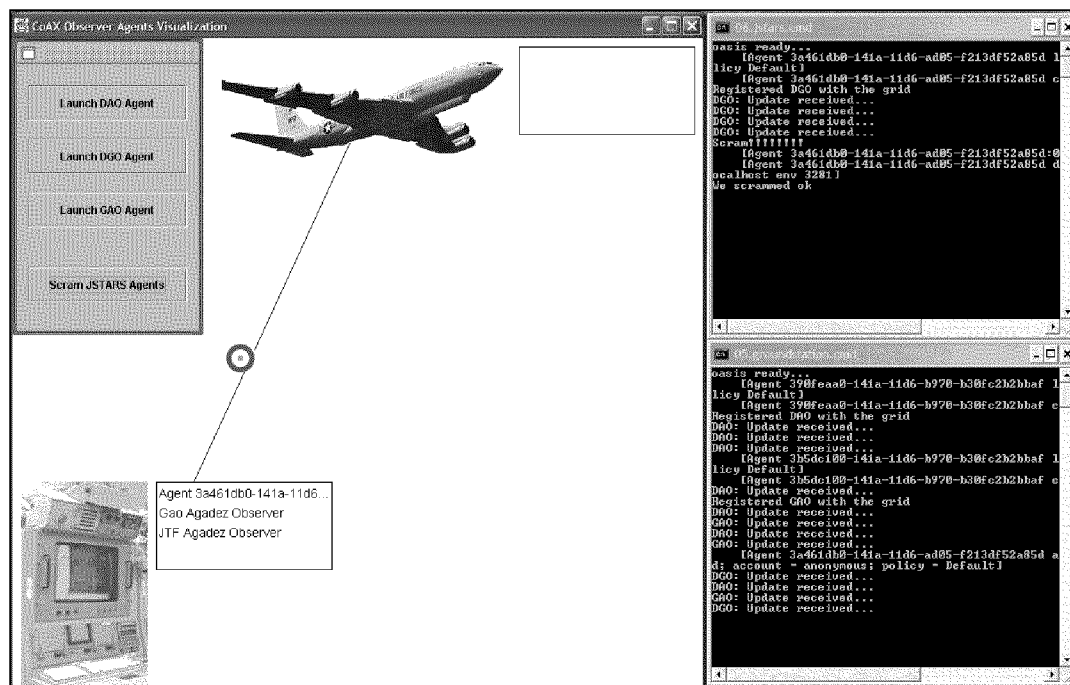


Figure 11: Forced migration of observer agents from mobile platform to ground station, using NOMADS and AromaVM. The updates from the DGO agent, initially on the JSTARS airborne platform (top right console) then start to appear on the new ground platform (lower right console).

5 Assessment of Software Agents

5.1 Technical Progress to Date

The CoAX project officially began in February 2000 and we believe that the demonstrations we have undertaken corroborate the hypotheses outlined in Section 1.3, demonstrating the utility of agent technology in Coalition operations. We have put together a prototype Coalition C2 architecture that supports and embraces heterogeneity and have exercised this in an agent-based C2 demonstration that enacts Coalition activities within the Binni scenario, including both the planning and execution phases of operations.

The CoABS Grid and KAoS domain management capabilities have allowed us to interoperate, for the first time, previously stand-alone US and UK military systems as well as a variety of agent-based information resources. In particular, the CoABS Grid has played a vital role in rapid and robust integration of systems. We have shown how agent organization, behavior, security and resources can be managed by explicit domain policy control.

Assessment work funded by the DARPA CoABS program has reported favorably on the performance issues of agent-enabled infrastructures and the experiences of the CoAX team have shown that the agent-wrapping of legacy systems and the integration of different agent systems at short notice is relatively straightforward. This task is simpler where systems expose more of their internal information and methods. In addition, a heterogeneous set of agents can be made to interoperate as long as implementers adhere to some minimum set of message and other standards. Heterogeneity should be accepted and embraced as it is seen as being inevitable and can actually be beneficial in a number of cases — especially in security terms.

Dynamic task, process and event handling is an important aspect of collaboration and Coalition C2. In the CoAX demonstrations a process panel was used to indicate the start of the tasking and lead into the heart of the demonstration. In the execution phase of operations, process panels in the main commands or headquarters were more extensively used as they enabled a clearer military relevant view of what was happening between the agents in less technical language than would otherwise be visible. Process and event panels have been found to be helpful in keeping users informed of the current stage of collaboration, and maintaining a shared picture of the current state of the collaborative efforts.

Our experience is that an agent-enabled environment gives the ability to create shared understanding and improved visualization. Specific benefits were gained when agents worked semi-autonomously in the background to process

information and support decision making collaboratively with operators, and when agents were integrated into existing tools so as not to disrupt familiar methods of operation.

5.2 Future Research Program

An aim only partially addressed in the current work is the construction and maintenance of a fully dynamic virtual Coalition organization. This would involve:

- domains and agents added to the Coalition structure ‘on-the-fly’;
- Coalition partners joining / leaving unpredictably;
- handling of dynamic Coalition tasks, processes and events.

Capabilities under investigation for future demonstrations include

- obligation management, e.g. ensure that agents are meeting their commitments;
- improved agent collaboration and run-time interoperability achieved using semantic web languages and technology (Allsopp et al, 2001a);
- richer domain organization and security policies (Bradshaw et. al., 2001);
- richer task, process and event management with more dynamically determined agent relationships (Tate et al., 2002);
- a variety of agents providing new types of data, and data-processing capabilities such as threat classification and track prediction.

Aspects of this work will be included in the Fleet Battle Experiment-Juliet 2002, part of the Millennium Challenge joint integrating experiment.

5.3 Military Implications of the Results

The CoAX research program has shown how software agents can carry out tasks that enable interoperability between information systems and infrastructure services brought together in a ‘come-as-you-are’ Coalition.

In the experiments so far, the software agents operated in a number of roles. They have worked ‘in the background’ — through matchmaking, domain management, process management and other agent services — to find, establish and maintain the infrastructure, information and procedural links necessary to achieve and support interoperability in a dynamically changing environment. In addition, they have worked collaboratively with human operators, mediating requests for information and formatting and displaying the results almost transparently.

Thus an agent-enabled environment helps create shared understanding and improves the situational awareness of military commanders. Moreover, it could make a significant contribution to the aims of Network-Centric Warfare which is defined as follows: an approach to the conduct of warfare that derives its power from the effective linking or networking of the warfighting enterprise. It is characterized by the ability of geographically dispersed forces to create a high level of shared battlespace awareness that can be exploited via self-synchronization and other network-centric operations to achieve commander’s intent.

One early lesson has been that Cyberspace should not be seen just as an information pipe between humans — it is a Battlespace in its own right. This indicates that ‘Cyberspace Superiority’ should be obtained (as for any other part of the Battlespace) by ensuring that Coalition forces are able to act decisively through software agents acting on behalf of or mediating the actions of human users.

Dealing effectively with unpredictable changes — owing, for example, to the destructive activities of opponents or because of systems failing and services being withdrawn — is a typical Coalition problem where software agents could make a significant contribution. So far, we have shown that a software agent infrastructure is robust and, to some extent, is ‘self-healing’. Our aim is to investigate this further to show that software agents can provide agility, robustness, flexibility and additional functionality beyond that provided by the individual Coalition partners.

6 Concluding Remarks

The central hypothesis being investigated in CoAX is that the agent-based computing paradigm is a good fit to the kind of computational support needed in Coalition operations. The evidence so far confirms this view: we have shown a number of disparate agent systems working together in a realistic Coalition application and indicated the value of the agent-based computing paradigm for rapidly creating such agent organizations. Agents can usefully share, and manage access to, information across a stylized Coalition architecture.

Our conclusion is that software agents, together with agent-based infrastructures and services provided by the CoABS Grid and KAoS, could play a key role in supporting Coalition operations. We think that this technology will provide the ability to bring together and integrate systems quickly to aid in all aspects of Coalition operations, without sacrificing security and control. Our long-term goal is to use this technology in the creation, support and dynamic reconfiguration of virtual organizations — with Coalitions being an archetypal and timely example of an area where this technology is vitally needed.

Acknowledgements

The authors gratefully acknowledge all those who contributed to the CoAX project, including Mark Burstein, Thom Bartold, Maggie Breedy, John Carson, Jeff Cox, Brad Clement, Rob Cranfill, Jeff Dalton, Pete Gerken, Bob Gray, Arne Grimstrup, Paul T. Groth, Greg Hill, Heather Holmback, Renia Jeffers, Martha Kahn, Shri Kulkarni, John Levine, Jean Oh, Pradeep Pappachan, Shahrukh Siddiqui, Jussi Stader, and Andrzej Uszok. The various projects that participated in CoAX were sponsored by the Defense Advanced Research Projects Agency (DARPA) and managed by the U.S. Air Force Research Laboratory, except work by QinetiQ, which was carried out as part of the Technology Group 10 of the UK Ministry of Defence Corporate Research Programme. The US Government and the contributors' organizations are authorized to reproduce and distribute reprints for their purposes notwithstanding any copyright annotation hereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either express or implied, of DARPA, the US Government, the US Air Force Research Laboratory, the UK MoD, or the contributors' organizations.

References

- Alberts, D. S., Garstka, J.J., Hayes, R.E., Signori, D. A. (2001) "Understanding Information-Age Warfare", CCRP Publication Series, 2001. ISBN 1-893723-04-6
- Allsopp, D.N., Beautement, P., Bradshaw, J.M., Carson, J., Kirton, M., Suri, N. and Tate, A. (2001) "Software Agents as Facilitators of Coherent Coalition Operations", Sixth International Command and Control Research and Technology Symposium, US Naval Academy, Annapolis, Maryland, USA, 19-21 June 2001.
- Allsopp, D.N., Beautement, P., Carson, J. and Kirton, M. (2001a) "Toward Semantic Interoperability in Agent-based Coalition Command Systems", Proceedings of the First International Semantic Web Workshop, July 30-31, 2001, Stanford University, CA, USA, pp 209-228
- Bradshaw, J.M., Suri, N., Kahn, M., Sage, P., Weishar, D. and Jeffers, R. (2001) "Terraforming Cyberspace: Toward a Policy-based Grid Infrastructure for Secure, Scalable, and Robust Execution of Java-based Multi-agent Systems". IEEE Computer, 49-56, July 2001.
- Bradshaw, J.M., Dutfield, S., Benoit, P. and Woolley, J.D. (1997) "KAoS: Toward an Industrial-Strength Generic Agent Architecture," Software Agents, AAAI Press/The MIT Press, Cambridge, Mass., pp. 375-418.
- Burstein, M., Ferguson, G. and Allen, J. (2000) "Integrating Agent-Based Mixed-Initiative Control with an Existing Multi-Agent Planning System", Proceedings of the Fourth International Conference on MultiAgent Systems, Boston, MA, 2000.
- Clement, B.J. and Durfee, E.H. (1999) "Top-Down Search for Coordinating the Hierarchical Plans of Multiple Agents", Proceedings of the Third International Conference on Autonomous Agents, pages 252-259, May 1999.
- Emerson, T. and Burstein, M. (1999) "Development of a Constraint-based Airlift Scheduler by Program Synthesis from Formal Specifications", Proceedings of the 1999 Conference on Automated Software Engineering, Orlando, FL, September, 1999.
- Fraser, J. and Tate, A. (1995) "The Enterprise Tool Set — An Open Enterprise Architecture", Proceedings of the Workshop on Intelligent Manufacturing Systems, International Joint Conference on Artificial Intelligence (IJCAI-95), Montreal, Canada, August 1995.
- Jennings, N R. (2001) "An Agent-based Approach for Building Complex Software Systems". Communications of the ACM. Vol 44, No: 4, 35-41. April 2001.
- Knoblock, C. A., and Minton, S. (1998) "The Ariadne Approach to Web-based Information Integration", IEEE Intelligent Systems , 13(5), September/October 1998.

Rathmell, R.A. (1999) "A Coalition Force Scenario 'Binni — Gateway to the Golden Bowl of Africa'", Proceedings of the International Workshop on Knowledge-Based Planning for Coalition Forces, (ed. Tate, A.) pp. 115-125, Edinburgh, Scotland, 10th-11th May 1999.

Tate, A. (1996) "The <I-N-OVA> Constraint Model of Plans", Proceedings of the Third International Conference on Artificial Intelligence Planning Systems, (ed. Drabble, B.), pp. 221-228, Edinburgh, UK, May 1996, AAAI Press.

Tate, A., Dalton, J. and Levine, J. (1998) "Generation of Multiple Qualitatively Different Plan Options", Fourth International Conference on AI Planning Systems (AIPS-98), Pittsburgh, PA, USA, June 1998.

Tate, A., Dalton, J., and Stader, J. (2002) "I-P² — Intelligent Process Panels to Support Coalition Operations", in Proceedings of the Second International Conference on Knowledge Systems for Coalition Operations (KSCO-2002) (ed. Tate, A.), 23/24-Sep-2002, Toulouse, France.

© Copyright QinetiQ Ltd 2002